

# Converging Work-Talk Patterns in Online Task-Oriented Communities

Qi Xuan<sup>1,2</sup>, Premkumar T Devanbu<sup>1</sup>, Vladimir Filkov<sup>1</sup>

{qxuan@, devanbu@cs., filkov@cs.}ucdavis.edu

<sup>1</sup>Department of Computer Science, University of California, Davis, CA 95616, USA

<sup>2</sup>Department of Automation, Zhejiang University of Technology, Hangzhou 310023, China

## Abstract

Much of what we do is accomplished by working collaboratively with others, and a large portion of our lives are spent working and talking; the patterns embodied in the alternation of working and talking can provide much useful insight into task-oriented social behaviors. The available electronic traces of the different kinds of human activities in online communities are an empirical goldmine that can enable the holistic study and understanding of these social systems. Open Source Software projects are prototypical examples of collaborative, task-oriented communities, depending on volunteers for high-quality work. Here, we use sequence analysis methods to identify the work-talk patterns of software developers in these online communities.

We find that software developers prefer to persist in same kinds of activities, i.e., a string of work activities followed by a string of talk activities and so forth, rather than switch them frequently; this tendency strengthens with time, suggesting that developers become more efficient, and can work longer with fewer interruptions. This process is accompanied by the formation of community culture: developers' patterns in the same communities get closer with time while different communities get relatively more different. The emergence of community culture is apparently driven by both "talk" and "work". Finally, we also find that workers with good balance between "work" and "talk" tend to produce just as much work as those that focus strongly on "work"; however, the former appear to be more likely to continue to be active contributors in the communities.

## 1 Introduction

A great deal of adult life is spent working. We work to create materials that fulfill human needs, to develop advanced technologies, to govern, heal, and teach each other, etc. Our work is often collaborative, and often involves repeated activities: i.e., we commute, work, collaborate with others, etc. Collaborations involve both *talking* and *working*. We get some work done, talk with our colleagues to socialize, learn, or further co-ordinate tasks, and then work some more. The recurrent practices constitute patterns of activities that can be used to characterize individuals, cluster them, and then predict their future behaviors; this has potential applications in various areas including crime control [35], traffic forecasting [33], and marketing [10]. In this paper, we will focus on the two most basic activities, i.e., work and talk. Note that, talking or communication, as important markers of human relations, play key roles in the coordination between pairs of co-operating individuals. As a result, they are commonly used to infer the social networks as the discrete spaces to study the dynamics of many other activities [40, 41]. Here, however, we treat work and talk activities equally, and use sequence analysis methods to reveal the work-talk (W-T) patterns of the individuals in online task-oriented communities.

Sequence analysis, which has long history of being useful in molecular biology [53], has been, as of recently, also used in social science [1, 13], where researchers investigate life courses [5], and career trajectories [2]. Whereas DNA sequences are curled up in three-dimensional space, social events are arranged according to their time of occurrence. Due to our interest in social phenomena mostly local in time, the positions of social events in a sequence refer to relative,

rather than absolute, time points. In bioinformatics, a number of global and local sequence alignment methods are used to compare the molecules’ genetic similarity within and across different organisms, so as to elucidate their biological functions [45, 47]. Here we adopt a local alignment method to find and enumerate short patterns in work-talk (W-T) sequences of different software developers. We use these short W-T pattern counts as data points for modeling developer behavior using hidden Markov models (HMMs) [21]. The goodness of fit of these models are established via their ability to predict the numbers of larger patterns in the sequences [47].

In collaborative projects the interplay of work and talk activities play a central role; therefore, the fitted parameters of these W-T HMMs can be used to characterize not just the W-T patterns of different individuals, but also help describe the projects’ work culture. By “work culture” here we mean the tendency of a group of individuals to share similar W-T patterns. The simplest such distinguishing pattern is that they either tend to work continuously (thus creating shared work products) or talk continuously (co-ordinating work with others, and strengthening relationships). More complex patterns are a combination of the two. This connotation of “culture” is consistent with Etzioni’s notion [23]: “the set of assumptions shared by members of a societal unit which sets a context for its view of the world and itself”. It is known that community culture plays an important role in innovation [30], the quality of work-products [6], and can facilitate the decision-making [16]. Community size and its evolution are related to productivity and efficiency. Recent work on collaboration indicate that team size [55] and the team assembling mechanisms [26] have significant effects on team performance.

We used open-source software (OSS) communities [20, 39, 54] to study W-T patterns for three main reasons. First, the work in OSS communities is easy to observe, and most of the talk activities are meaningfully related (because of community norms) to work activities; this simplifies the observation of functional W-T patterns. Second, the work and talk activities in OSS communities are always archived [42], so they are readily collected for analysis. Finally, performance properties, such as productivity, in terms of number of lines of code written, can also be measured using

the state of the produced software. Communication is vital in making collaboration effective [57]. Lack of communication between software developers may introduce more coordination problems [28], and thus the distributed developers do need to maintain awareness of one another to make the OSS projects successful [27]. A recent empirical research indicates that communication and commits may accelerate each other in Apache OSS projects [58], and similarly in Stack Overflow and GitHub [50]. However, it is also argued that, since both communication and working activities may compete for the time resources of individuals, communication may have some negative effect on the efficiency of work activities [38, 59]. In this case, communication can be considered as a kind of interruptions, recovering from which developers may take some time to continue their work [36, 49].

In this paper, by studying W-T patterns of developers in OSS communities, we make the following main contributions.

- We establish HMM models for the W-T sequences of developers and find the evidence of community culture: developers in the same community tend to have more similar W-T patterns than those from different communities, and this pattern-affinity strengthens with time.
- We observed that developers who have balanced W-T patterns are just as productive as those who work more continuously (fewer interruptions), but the former tend to stay active longer; this suggests that W-T balance is important to sustain OSS communities.
- We create social and cooperation networks, and find that the convergence of W-T patterns between a given pair of developers appears to be re-inforced by both talk- and work-related interactions. This indicates that the emergence of a community W-T culture is apparently driven by both “work” and “talk”, and thus offers a novel perspective on the co-evolving mechanisms of socio-technical, inter-dependent networks [12, 14, 56].

Project	Description	Time frame	#Users	#Devs	#S-Devs	#Files
Activemq	Integration patterns server	2005/12/12-2012/03/16	2012	28	6	16788
Ant	Build tool	2000/01/13-2012/03/16	1402	44	9	11620
Axis2_c	Web services engine	2004/02/03-2012/03/15	582	24	8	10262
Axis2_java	Web services engine	2001/01/30-2012/03/19	3738	72	15	129978
Camel	Integration framework	2007/03/19-2012/03/17	805	31	6	36965
Cxf	Web services framework	2005/07/22-2012/03/16	427	45	7	37867
Derby	Database management system	2004/08/10-2012/03/22	1118	35	16	6563
Lucene	Search software	2001/09/11-2012/03/23	2102	41	14	6674
Mahout	Machine learning library	2008/01/15-2012/03/23	533	15	6	5123
Nutch	Web search software	2005/01/25-2012/03/22	556	16	6	3072
Ode	Web services	2006/02/18-2012/03/22	365	17	6	11006
Openejb	Container system and server	2002/01/18-2012/03/22	169	38	5	43960
Solr	Enterprise search platform	2006/01/20-2011/03/01	825	19	8	8534
Wicket	Web application framework	2004/09/21-2012/03/21	539	24	8	48045

Table 1: Basic properties of the fourteen OSS communities.

## 2 Work-Talk Activities in OSS

Human activities can be represented by time-series denoting the event occurrences. As a generalization, different kinds of activities can be represented by an asynchronous multiple time-series [37] since people seldom do different things at exactly the same time; this is quite different from the multiple time-series in economics where different kinds of indexes are recorded at the same time for comparison. Since work and talk are the activities of concern here, we will use sequence analysis to study these two kinds of activities in OSS communities, with primary focus on their *relative time orders*, without concern for the *precise time of occurrence*.

### 2.1 Data Description

We collected the individual work and talk activities from 31 OSS developer communities in the *Apache Software Foundation* on March 24th, 2012. In each community, there are several volunteer developers who contribute by committing to files, i.e., adding or removing software code; these activities are recorded in a Git repository. These are our “work events”, or W. Members of the online communities communicate using the *developer mailing lists* with the rest of the community through emails to share programming knowledge, and coordi-

nate with others. We record the sent emails as “talk events”, or T, of a developer (the received emails are included in the talk activities of others). Using this data, a W-T sequence concerning the work and talk activities can be recorded for each developer. Note that messages may be automatically posted to a mailing list in an OSS community to inform others when some work is done. In order to exclude such trivial talk activities, here we just consider those response emails [58] which takes up about 73% of all messages [8]. Moreover, we also use a semi-automatic approach to solve the problem of multiple aliases[8].

We pre-process the W-T sequence data in several ways. To ensure a sufficient number of samples to reliably compare the W-T patterns between the pairs of developers of the same or from different communities, we select a subset of the developers with sequences including at least 500 work and talk activities, and a subset of communities with at least 5 such developers. We acknowledge a risk of left-censorship of both work & talk activities, if any OSS communities did not archive their emails, or if they had used different version control systems before they moved to Git, some early data could be lost. Besides, it is known that many individuals need to first earn social capital in the OSS project by communicating with others before they are accepted as developers [9, 25]. As a result, we often observe long,

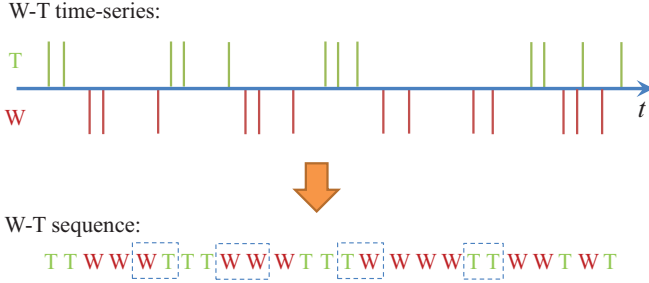


Figure 1: A multiple time-series of work and talk activities and the corresponding W-T sequence. The four different two-patterns, i.e., WW, WT, TW, and TT, are marked by the dashed rectangles.

pure work (or talk) subsequences before the first talk (or work) activity of a developer. In this study, we remove these trivial prefixes of pure work or talk activities, i.e., we only consider W-T sequences starting from the first work (or talk) activity if it occurred after the first talk (or work) activity.

After pre-processing the data, we are left with 14 communities totally containing 120 developers, and several of their basic properties are presented in Table 1. Note that besides the developers, we also list the number of active users (including developers) in each project. These users might not directly commit to files, but they may contribute to the communities by other ways, such as reporting bugs etc.

## 2.2 Pattern Analysis

A  $G$ -pattern in a sequence over the alphabet  $\{W, T\}$  is a subsequence of length  $G$ . There are total  $2^G$  possible different  $G$ -patterns. Typically, the length of the patterns is much shorter than the length of the given sequence. In our study we focus on 2-patterns and 3-patterns. Given a sequence  $\theta = \{s_1, s_2, \dots, s_h\}$  over  $\{W, T\}$ , we count the occurrence of each of the  $2^G$  patterns, by rolling a window of size  $G$  over the sequence, and incrementing the count for the pattern we find. E.g., in the W-T sequence shown in Figure 1, the four possible 2-patterns, WW, WT, TW, and TT, occur eight, five, five, and six times, respectively.

To assess the probability that a pattern occurs by chance, we create a null (baseline) model by randomizing the observed W-T sequence so as to preserve the pro-

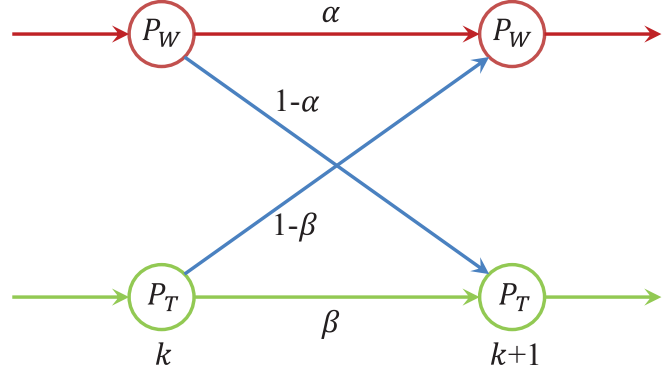


Figure 2: An HMM with two states, i.e., “work” and “talk”, denoted by “W” and “T”, respectively. The model is used to explain the W-T patterns of developers in different communities.

portion of work to talk activities. This can be achieved, e.g., by using the R package’s [48], sample() function, on the sequence indexes. Then, the preference for pattern  $P$  in the observed sequence,  $\theta$ , over the randomized sequence,  $\theta^*$ , is calculated by the relative difference between the counts for that pattern,  $C_P$  and  $C_P^*$ , in the respective sequences,

$$\lambda_P = \frac{C_P - \langle C_P^* \rangle}{\langle C_P^* \rangle} \times 100\%. \quad (1)$$

For each pattern  $P$  in a sequence, we also calculate its Z-score [32] as  $\lambda_P \langle C_P^* \rangle / \varsigma$ , where  $\varsigma$  is the standard deviation of the pattern counts in  $\theta^*$ . For  $\langle C_P^* \rangle$ , we generated 100 randomized sequences for each observed one, and the absolute values of the Z-scores for 2-patterns in our study are larger than 5, in 462 out of 480 cases, indicating that the observed counts are surprising. These random sequences are also used as references to evaluate the HMM model (represented in Figure 2) on predicting 3-patterns. (See the Appendix for further mathematical description of the HMM.)

## 3 Work-Talk Patterns are Conserved over Short Time Intervals

A work-talk pattern is a string, or word, comprised of the letters W and T, signifying the work and talk activities. The simplest non-trivial pattern is of length two,

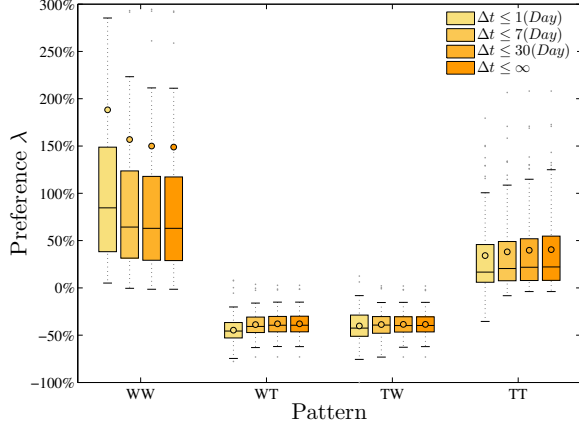


Figure 3: The box-and-whisker diagram for the preferences of the four different two-patterns in the real W-T sequences under the different time-interval conditions by comparing with the random ones.

which considers two successive activities (same or different) at a time. There are four possible different two-patterns: work-work (WW), work-talk (WT), talk-work (TW), and talk-talk (TT). Given an observed W-T sequence for each person, we count in it the occurrences of two-patterns. Then, we derive the preference of each pattern, denoted by  $\lambda_i, i = 1, 2, 3, 4$ , respectively, in the real sequences as compared to random ones (for each real sequence we bootstrapped from it 100 simulated W-T sequences, by randomizing the order of its elements). In our data set, we find that, on average for all developers,  $\lambda_1 = 148.9\%$  and  $\lambda_4 = 40.5\%$ , while  $\lambda_2 = -38.0\%$  and  $\lambda_3 = -38.6\%$ , i.e., WW and TT are positively enriched, while WT and TW are negatively enriched. This suggests that developers much prefer to persist with one activity-type, rather than switch between activities.

It may be argued that two successive activities should not be considered as a two-pattern if the time interval between them is relatively long, e.g., longer than one month. To show that our method is robust with respect to time-scale, we also calculate the relative difference by varying the thresholds for the time-intervals over which we consider the two-patterns. We vary the thresholds, denoted by  $\xi = 1, 7, 30$  (days), and only the patterns with intervals  $\leq \xi$  are considered. The results are shown

in Figure 3, where we can see that WW and TT patterns are always much more preferred than WT and TW patterns in the real sequences under thresholds varying from one day to one month. Interestingly, we also find a slight trend that the WW pattern becomes more preferred, and the TT pattern less preferred, when we exclude more repeated activities with relatively shorter time intervals (and thus a smaller  $\xi$ ). Since the number of these long time-interval patterns is relatively small (2.2% and 0.3% for  $\xi = 7$  and  $\xi = 30$ , respectively), this slight trend still indicates that developers are more likely to start and end a repeated and relatively compressed work sequence with talk activities, viz., talk activities plays important role in enabling new tasks (work activities) in these online communities.

In order to study W-T patterns in more detail, we create a two-state HMM with parameters  $\alpha$  and  $\beta$  representing the conditional transition probabilities  $P(W|W)$  and  $P(T|T)$ , respectively, for each developer (see the Appendix). Based on the model, we have

$$\alpha = \frac{P_{WW}}{P_{WW} + P_{WT}}, \quad \beta = \frac{P_{TT}}{P_{TT} + P_{TW}}, \quad (2)$$

where  $P_{WW}$ ,  $P_{WT}$ ,  $P_{TW}$ , and  $P_{TT}$  denote the probabilities of the four different two-patterns for each developer, and can be estimated by the counting numbers of the four different two-patterns, i.e.,  $M_i, i = 1, 2, 3, 4$ , respectively. Then, we get

$$\alpha = \frac{M_1}{M_1 + M_2}, \quad \beta = \frac{M_4}{M_3 + M_4}, \quad (3)$$

as long as the corresponding W-T sequence contains enough elements. Here, it is not difficult to prove that the condition  $|M_2 - M_3| \leq 1$  must be satisfied for any W-T sequence.

This HMM is fully determined by the numbers of the four different two-patterns. We validate the model by checking its ability to predict the numbers of larger patterns, e.g., three-patterns. There are eight different three-patterns, WWW ... TTT. For each developer, we denote by  $M_i$ ,  $M_i^*$ , and  $M_i^\circ$ ,  $i = 1, 2, \dots, 8$ , the numbers of three-patterns in the real sequence, the random sequence, and the sequence created by the model (with the same length and the initial element), respectively. Then, for this developer, we can calculate the relative error introduced by the random mechanism and the model

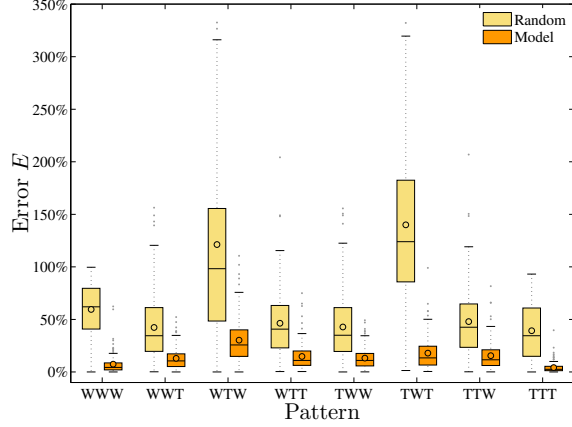


Figure 4: The box-and-whisker diagram for the relative errors of the eight different three-patterns introduced by the random mechanism and the HMMs, comparing with the real ones.

by

$$E_i^* = \frac{|M_i^* - M_i|}{M_i}, \quad E_i^o = \frac{|M_i^o - M_i|}{M_i}, \quad (4)$$

respectively. For each developer, and for each three-pattern, the difference between these two errors can be used to validate the ability of the HMM in predicting that pattern: viz., if the relative errors introduced by the model is significantly smaller than those introduced by the random mechanism, it is reasonable to believe that the model feasibly predicts that pattern. The differences between these two kinds of errors for all the eight different three-patterns are visualized by the box-and-whisker diagrams shown in Figure 4 with all the developers considered together, where we can see that the HMM does indeed predict the numbers of all the eight three-patterns with significantly smaller relative errors ( $p = 1.8 \times 10^{-16}$  on average) than the random mechanism for the developers we studied, i.e., 14.5% versus 67.4% on average.

## 4 Community Culture

Generally, people sharing similar habits and interests are more likely to come together; once together, they could further influence each other, so as to form the commu-

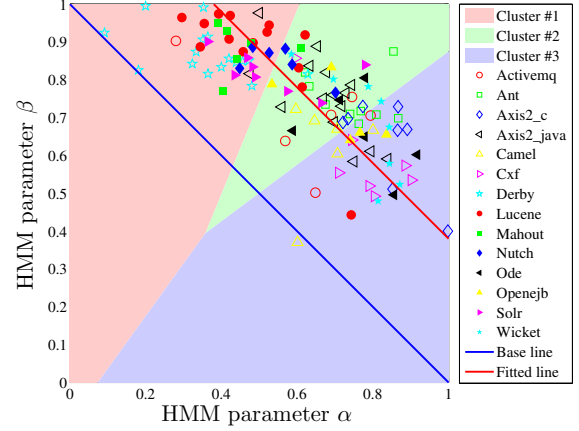


Figure 5: Visualization of developers on  $\alpha$ - $\beta$  plane by considering their whole sequences, where developers are point and those of the same communities are marked by the same symbols. The parameters are grouped into three clusters by the “K-means” method. The base line is formed by the HMM parameters of the random W-T sequences with different fractions of work activities. The points are fitted by the linear function  $\alpha + \beta = \varepsilon$ , with  $\varepsilon = 1.38$ .

nity culture. Then, one interesting question is: do developers in the same OSS communities present more similar W-T patterns (or closer  $\alpha$  and  $\beta$  more specifically) than developers from different communities, i.e., can W-T patterns be used as a metric to characterize community culture?

### 4.1 Converging Work-Talk Patterns and Community Culture

In order to answer the above question, we first visualize all  $(\alpha, \beta)$  pairs in the  $\alpha - \beta$  plane, as shown in Figure 5, where the developers of the same communities are marked by the same symbols. Evidence of clustering is visually apparent: the points representing the developers in the same communities are indeed closer to each other when compared with those from different communities. We further classified all the developers into three groups by the k-means method [31], and find that most developers in the same communities are centralized in one of three clusters, rather than uniformly distributed in all the three, which indicates different community cultures

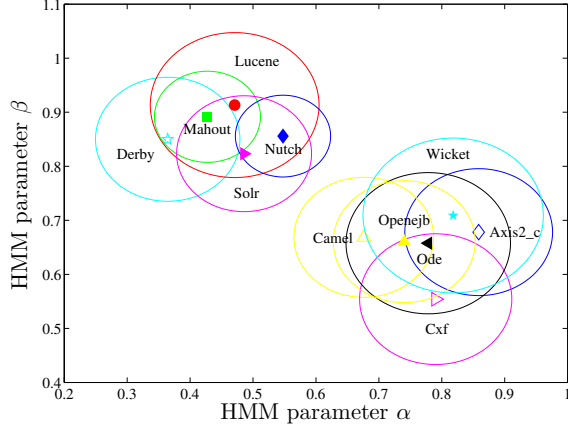


Figure 6: The centers and the respective diversities (the large circles) of the eleven communities on  $\alpha - \beta$  plane, defined as the medians of the HMM parameters of their developers and the average distances of HMM parameters between the developers and the corresponding centers, respectively.

that emphasize continuous work (cluster #3), talk (cluster #1), or both (cluster #2), respectively. Here, we also provide the baseline formed by the HMM parameters of the W-T sequences that are generated by the random mechanism with different fractions of work activities. Since this baseline must satisfy  $\alpha + \beta = 1$ , and almost all the points being above this based line validates again the preferred patterns WW and TT in all communities.

More specifically, most developers ( $\geq 50\%$ ) in *Derby*, *Lucene*, *Mahout*, *Nutch*, and *Solr* belong to cluster #1, which corresponds to mostly talk activities (high  $\beta$ ), while most of the developers in *Axis2\_c*, *Camel*, *Cxf*, *Ode*, *Openejb*, and *Wicket* belong to cluster #3, corresponding to mostly work activities (high  $\alpha$ ). As a whole, we define the center of a community in  $\alpha - \beta$  plane by the median of the HMM parameters of the developers in it, then calculate its diversity by the average distances of HMM parameters between the developers and the center, as shown in Figure 6 for the above 11 communities. It is interesting to find that the communities sharing similar W-T patterns also belong to similar domains (description in Table 1). For example, *Lucene*, *Nutch*, and *Solr* are all about “search” and they are intrinsically related to each other, just like the introduction of

*Nutch* on its web site: “Stemming from Apache *Lucene*, Apache *Nutch* now builds on Apache *Solr* adding web-specifics”. Besides, *Axis2\_c*, *Cxf*, and *Ode* are all about “services”, while each of *Camel*, *Cxf*, and *Wicket* is a software framework that provides a shared architecture for class of applications.

More formally, if we denote by  $\alpha_i$  and  $\beta_i$  the HMM parameters of developer  $d_i$ , we can calculate the Euclidean distance of HMM parameters between two developers  $d_i$  and  $d_j$  by

$$\rho_{ij} = \sqrt{(\alpha_i - \alpha_j)^2 + (\beta_i - \beta_j)^2}, \quad (5)$$

as a quantitative metric for the similarity between the W-T patterns of developers, i.e., the shorter the distance between them, the more similar the W-T patterns of the two developers. Then, we compare the distances of HMM parameters between all pairs of developers in the same communities with those between pairs of developers from different communities, and find that the former list of distances are significantly shorter ( $p = 0$ ) than the later ones. These qualitative and quantitative analysis lend support to using the HMM parameters as a reasonable proxy for the way the interplay of work and talk testify to community culture.

The clustering phenomenon of W-T patterns gives rise to another interesting question: Do developers choose to join communities with similar W-T patterns as theirs or does the similarity emerge over time as developers participate and evolve with their communities? In the first case, the developers in the same community will present similar W-T patterns from the very beginning, while in the second case, they will get more similar with time. To answer this question, we do the same pattern analysis as above, using only the initial 100 activities in the W-T sequences. Based on the comparison, we find that:

1. The developers in the same community showed similar W-T patterns starting with their inception into the project. I.e., for their first 100 activities, the distances of HMM parameters between pairs of developers in the same communities are significantly shorter ( $p = 3.1 \times 10^{-13}$ ) than those from different communities.



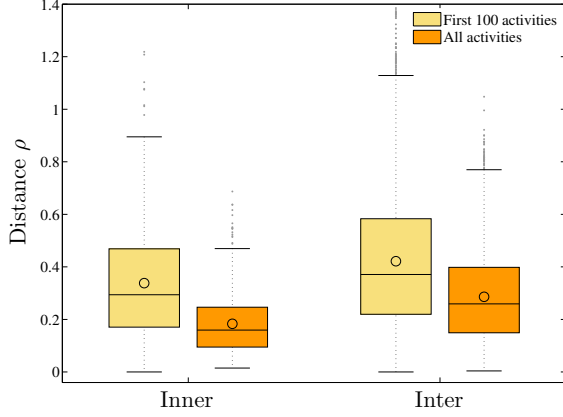


Figure 7: The box-and-whisker diagrams for the distances of the HMM parameters of the first 100 activities and those of the whole W-T sequences between pairs of developers inner and inter communities.

2. In addition, the community cultures of different communities converge rather than diverge from each other, as time evolves. I.e., both the inner (within-community) and inter (between-community) distances decrease significantly ( $p = 0$ ) with time, as shown in Figure 7. We also calculate the average inner distance for all communities by considering their respective first  $\varrho$  activities with different values of  $\varrho$ , as shown in Figure 8, to study the converging process. We find that the inner distances decrease as  $\varrho$  increases, for most communities. As examples, the evolutions of the HMM parameters with time for the communities *Axis2\_java*, *Derby*, and *Lucene* are shown in Figure 9.

3. The clustering of the HMM parameters within communities grows tighter with time. I.e., the convergence rates of the parameter distances from the first 100 activities to all activities within communities (the average distance decreases from 0.3381 to 0.1832) is significantly larger ( $p = 1.7 \times 10^{-7}$ ) than those between communities (it decreases from 0.4216 to 0.2861).

These findings suggest that developers with similar W-T patterns are indeed more likely to join in the same communities, and continue to harmonize their patterns

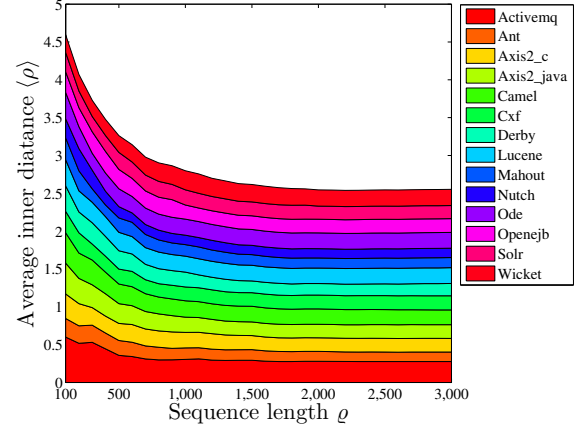


Figure 8: The average inner distances of HMM parameters between pairwise developers for the fourteen communities.

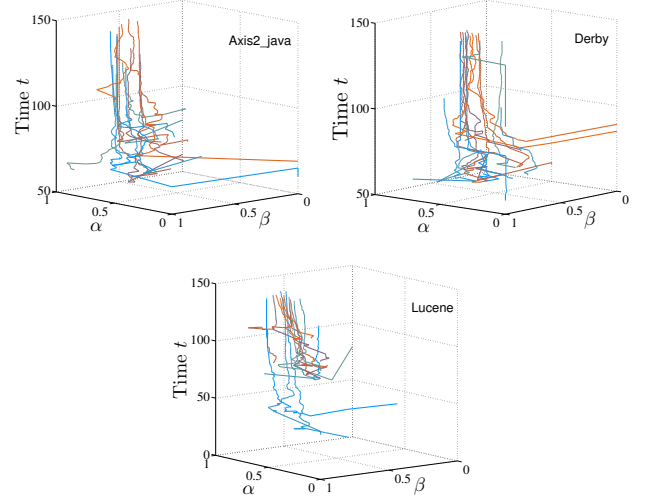


Figure 9: Developers'  $\alpha$  &  $\beta$  monthly evolving curves, e.g., *Axis2\_java*, *Derby*, and *Lucene*.

as they work and talk as a team. In fact, since there are many online communities on similar topics, people can first experience the culture of these communities and then decide to join or not [17, 43, 44]. For OSS, it is clear that most developers do communicate a fair bit on the developer mailing list before actually contributing work [9, 52]; indeed, this type of “socialization” is a necessary pre-requisite to having one’s work contributions accepted. Thus, it is to be expected that the de-



velopers are more likely to join in the communities with harmonized work and talk patterns, in order to reduce co-ordination efforts.

In addition, we find that different community cultures will slightly converge rather than diverge from each other over time; this suggests that there may be an overarching trend of the W-T patterns for all the developers (in all projects). To investigate this further, we compare the two parameters  $\alpha$  and  $\beta$  separately for all developers, considering *a*) the first 100 activities and *b*) all activities. We find that both of them increase as time evolves, i.e., the HMMs in case *a*) have significantly smaller  $\alpha$  ( $p = 2.7 \times 10^{-2}$ ) and  $\beta$  ( $p = 1.4 \times 10^{-5}$ ) than those in *b*). In fact, the efficiency of overall work and talk activities may be measured by the sum  $\alpha + \beta$ ; larger values of this sum indicate less switching between activities and thus fewer interruptions. This arguably represents higher efficiency [4, 7, 36]. In other words, the HMM parameters  $(\alpha_i, \beta_i)$  shown in Figure 5 can be fitted by the linear function:

$$\alpha + \beta = \varepsilon, \quad (6)$$

with a single parameter  $\varepsilon$  representing the average efficiency of all the developers. Using the least squares method, we get the average efficiency  $\varepsilon$  and the corresponding standard deviation  $\sigma$  from the regression line as

$$\varepsilon = \frac{\sum_{i=1}^N (\alpha_i + \beta_i)}{N}, \quad \sigma = \sqrt{\frac{\sum_{i=1}^N (\alpha_i + \beta_i - \varepsilon)^2}{2N}}, \quad (7)$$

respectively, for the  $N$  developers. We find that the average efficiency steadily increases, while the variance decreases, with time, which means that as time goes on developers tend to have longer bursts of pure work and pure talk, suggesting that their discussions are becoming more effective, and that the ensuing co-operative work proceeds relatively more uninterruptedly.

Looking at the change in the rate of talk activities for all developers, in terms of  $\alpha$  and  $\beta$ , equation (16), we find that the rate increases significantly ( $p = 4.6 \times 10^{-3}$ ) with time, indicating that most developers become more socialized in the process. This phenomenon is consistent with the fact that more discussions are always needed to further improve a mature product. Meanwhile, contributing to these online communities is social work, i.e.,

the contributions of developers are highly visible and will be checked by many other users, so it is not surprising that they need to reply to comments more frequently when contributing more.

## 4.2 Individual Performance and Community Culture

Regarding community culture, it is always very interesting to try to identify its benefits on individuals in the respective communities. For example, it is reasonable to hypothesize that developers who work more than they talk will have higher productivity, meaning they will produce more lines of codes (LoC), than those seeking balance between work and talk activities (similarly, those preferring talk over work may have a socialization advantage compared to those seeking balanced activities). We ask, does increasing productivity always come at the price of decreasing socialization, and vice versa? Note that looking simply for strong work or talk preference, i.e., larger  $\alpha$  or  $\beta$ , respectively, does not necessarily lead to higher productivity or socialization because our sequence analysis does not take the activity time into consideration and is also independent from the length of the sequences.

To answer that we study the correlations with community culture of five measures of individual performance *work rhythm* (# work activities per day), *thousands of lines of code added per unit time* (KLoC), *talk rhythm* (# talk activities per day), *newly established social links per week*, and *observed survival time*, resp.,  $X_1$  to  $X_5$ , as summarized in Table 2. The first four properties are calculated in the same time period of the person's W-T sequence. The *survival time*,  $X_5$ , of a developer is defined as the period of time from their first activity to the last one, which may be longer than the period of their W-T sequence, considering that the W-T sequences under study were preprocessed by removing prefixes of pure work or talk activities. The survival time of a developer is only observed when the developer has left the respective community. Here, as a reasonable estimation, we consider that a developer has left the community if they have not been active for a relatively long time, i.e., longer than some threshold  $T$ .

All developers are divided into three clusters by their HMM parameters, as shown in Figure 5. The developers

Property	Description	Inter cluster T-test ( $p$ )		
		#1 vs. #2	#2 vs. #3	#1 vs. #3
$X_1$	Working rhythm: the number of work activities per day	$1.9 \times 10^{-2}$	$5.4 \times 10^{-3}$	$1.2 \times 10^{-7}$
$X_2$	The kilo lines of added codes (KLoC) per day	$1.5 \times 10^{-2}$	$7.8 \times 10^{-1}$	$3.3 \times 10^{-2}$
$X_3$	Talking rhythm: the number of talk activities per day	$1.2 \times 10^{-2}$	$5.8 \times 10^{-2}$	$6.6 \times 10^{-6}$
$X_4$	The number of new social links per week	$9.0 \times 10^{-1}$	$3.7 \times 10^{-2}$	$2.0 \times 10^{-3}$
$X_5$	The observed survival time (year)	$3.0 \times 10^{-1}$	$3.7 \times 10^{-2}$	$2.7 \times 10^{-1}$

Table 2: The student’s t-tests for five individual properties between different clusters.

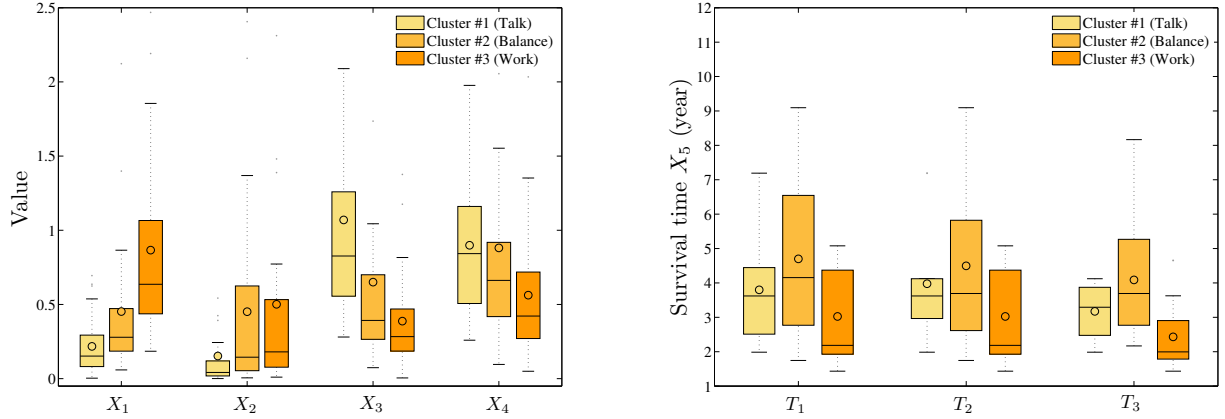


Figure 10: The effects of community culture on individual properties. The box-and-whisker diagrams for (left) the four individual properties  $X_1$  to  $X_4$ , and (right) the observed survival time  $X_5$  with different time thresholds  $T_1$  (half year),  $T_2$  (one year), and  $T_3$  (two years), for the developers in the three clusters determined by their HMM parameters.

in Cluster #1 emphasize “talk”, those in Cluster #3 emphasize “work”, while those in Cluster #2 seek balance between the two. For each property from  $X_1$  to  $X_4$ , we have a list of their values for developers in each cluster, and the comparisons between the properties of developers in different clusters are visualized by the box-and-whisker diagrams shown in Figure 10 (left), with the significance presented in Table 2. We find that the developers in Cluster #3 have the fastest working rhythms, those in Cluster #2 follow, while the developers in Cluster #1 work the slowest. The direction reverses for their talking rhythms. However, the situation is a little different when we compare the abilities of developers of different clusters in producing codes and earning social status. We find that the developers in Cluster #2 and Cluster #3 can produce similar KLoC per day, and both groups produce significantly more than the developers in

Cluster #1, while the developers in Cluster #2 and Cluster #1 earn similar numbers of social links per week, and both groups earn significantly more than the developers in Cluster #3. These indicate that extended discussion is always accompanied with the slowing down of work rhythms, but not always with decrease of productivity, and the developers seeking balance between work and talk behave competitively on both productivity and socialization as those who mostly work or mostly talk.

Although it seems that the developers who mostly work have the fastest working rhythms and the highest productivity, on average, it doesn’t mean that choice is the healthiest for them or for the overall project, since these developers are more likely to feel boring and then quit the projects. To analyze the survival times of developers (time from joining until leaving) in terms of the HMM parameters  $\alpha$  and  $\beta$ , we use the Hazard

model [22], with the Hazard ratio defined as

$$\eta = e^{bx}. \quad (8)$$

where  $x$  is either  $\alpha$  or  $\beta$ . (See the Appendix for further mathematical descriptions of the Hazard model.) We find that developers with smaller  $\alpha$  or larger  $\beta$  will have suggestively longer survival times ( $p = 0.077$  and  $b = 1.7$  for  $\alpha$  and  $p = 0.042$  and  $b = -2.4$  for  $\beta$ ), indicating that, by comparison, talk activities are more important than work activities for developer retention. Indeed, we find that developers with more balance between their work and talk stay active in the projects for suggestively longer periods of time than those who mostly work, as shown in Figure 10 (right), i.e., the significance is equal to 0.037, 0.078, and 0.049 when the survival times of the developers with their last activities occurred half year, one year, and two years before are considered, respectively. The significance of comparison for the survival time among the three clusters of developers are presented in Table 2 when  $T = 1$  (year). These findings suggest that developers with balanced W-T patterns are important to sustain OSS communities. Each of the communities we studied has at least one balanced developer, and there is also a natural trend that developers become more balanced, i.e., both  $\alpha$  and  $\beta$  increase with time.

## 5 Synchronizing Role of Social and Technical Links

It is well understood that individuals can influence each other's behaviors through social links [11, 15, 24]. Here, we study the extent to which developers with similar W-T patterns tend to be linked more in the email network or the technical cooperation network. In social networks, social weight between two developers intuitively means the number of emails between them. In cooperation networks, a pair of developers are linked with an edge indicating the number of files on which they have both worked. In particular, denoting by  $\psi_i$  the list of files that developer  $d_i$  commits to, the cooperative weight between a pair of developers  $d_i$  and  $d_j$ , in terms of the files to which they have committed, is defined as

$$\omega_{ij} = \frac{\psi_i \cap \psi_j}{\psi_i \cup \psi_j}. \quad (9)$$

On the social end, for pairs of developers, we ask: are the distances between their HMM parameters correlated with the number of emails they have exchanged? The results of using both Pearson and Spearman correlations are given in Table 3, under the Social weights columns. We find negative correlation in ten out of fourteen communities with both methods, with the significance  $p < 0.05$  in eight of them, while we find positive correlation with the significance  $p < 0.1$  in only one community called *Mahout*. The negative correlation means that the smaller the HMM parameter distance between two developers, the larger the number of emails they have exchanged.

On the technical end, we study the correlation between the distances of HMM parameters and strength of file cooperation links between developers. Using the same correlation measures as before, we get the results in Table 3, under the Cooperative weights columns. In this case negative correlation is found in eleven out of fourteen communities with both methods, with the significance  $p < 0.1$  in four of them, including *Activemq*, *Ant*, *Derby*, and *Solr*, while no community has positive correlation with significance  $p < 0.1$ . The negative correlation means that the smaller the HMM parameter distance between two developers, the larger the cooperation between them.

When considering all communities together, we obtain a significantly negative correlation for both methods in both cases (the last row of Table 3). Thus, developers with more emails between them or committing to more of the same files are more likely to have similar W-T patterns. The results also indicate that community culture may be either social or task oriented (technical); the distances between HMM parameters are more likely to be correlated with social weights in some communities, and with cooperative weights in others.

## 6 Threats to Validity

Although most results shown in this paper are significant, there are still a number of threats to this study.

All the OSS projects under study are collected from Apache, which may limit the generalization of the results. The methods therefore need to be tested on other OSS ecosystems, such as GitHub [19] and

Communities	Social weights				Cooperative weights			
	Pearson		Spearman		Pearson		Spearman	
	$R$	$p$	$R$	$p$	$R$	$p$	$R$	$p$
Activemq	-0.3287	$2.3 \times 10^{-1}$	-0.3056	$2.7 \times 10^{-1}$	-0.4427	$9.9 \times 10^{-2}$	-0.5607	$3.2 \times 10^{-2}$
Ant	0.0826	$6.3 \times 10^{-1}$	0.0049	$9.8 \times 10^{-1}$	-0.3753	$2.4 \times 10^{-2}$	-0.3704	$2.7 \times 10^{-2}$
Axis2_c	-0.4833	$5.8 \times 10^{-2}$	-0.4667	$1.2 \times 10^{-2}$	0.2209	$2.6 \times 10^{-1}$	0.2474	$2.0 \times 10^{-1}$
Axis2_java	-0.1611	$1.0 \times 10^{-1}$	-0.0442	$6.5 \times 10^{-1}$	-0.0797	$4.2 \times 10^{-1}$	-0.1714	$8.1 \times 10^{-2}$
Camel	-0.4896	$6.4 \times 10^{-2}$	-0.6000	$2.0 \times 10^{-2}$	-0.2424	$3.8 \times 10^{-1}$	-0.3679	$1.8 \times 10^{-1}$
Cxf	-0.0036	$9.9 \times 10^{-1}$	0.0651	$7.8 \times 10^{-1}$	0.0711	$7.6 \times 10^{-1}$	0.1948	$4.0 \times 10^{-1}$
Derby	-0.2258	$1.3 \times 10^{-2}$	-0.1940	$3.4 \times 10^{-2}$	-0.2430	$7.5 \times 10^{-3}$	-0.3232	$3.4 \times 10^{-4}$
Lucene	-0.3856	$1.6 \times 10^{-4}$	-0.6046	$2.2 \times 10^{-10}$	-0.1361	$2.0 \times 10^{-1}$	-0.2275	$3.0 \times 10^{-2}$
Mahout	0.6829	$5.0 \times 10^{-3}$	0.6685	$6.4 \times 10^{-3}$	0.1220	$6.7 \times 10^{-1}$	-0.3429	$2.1 \times 10^{-1}$
Nutch	-0.2265	$4.2 \times 10^{-1}$	-0.2832	$3.1 \times 10^{-1}$	0.3648	$1.8 \times 10^{-1}$	0.4071	$1.3 \times 10^{-1}$
Ode	-0.4722	$7.6 \times 10^{-2}$	-0.4866	$6.6 \times 10^{-2}$	-0.1250	$6.6 \times 10^{-1}$	-0.1429	$6.1 \times 10^{-1}$
Openejb	-0.0017	1.0	0.0667	$8.6 \times 10^{-1}$	-0.6465	$4.3 \times 10^{-2}$	-0.3818	$2.8 \times 10^{-1}$
Solr	-0.3188	$9.8 \times 10^{-2}$	-0.5083	$5.8 \times 10^{-3}$	-0.5426	$2.9 \times 10^{-3}$	-0.5457	$3.1 \times 10^{-3}$
Wicket	-0.2512	$2.0 \times 10^{-1}$	-0.1363	$4.9 \times 10^{-1}$	-0.2326	$2.3 \times 10^{-1}$	-0.1795	$3.6 \times 10^{-1}$
All	-0.2396	$1.6 \times 10^{-8}$	-0.2517	$2.8 \times 10^{-9}$	-0.1420	$9.1 \times 10^{-4}$	-0.2037	$1.8 \times 10^{-6}$

Table 3: The Pearson & Spearman correlation tests for the distances of HMM parameters and social & cooperative weights between pairwise developers for different communities.

GNOME [46], or other online volunteer communities, such as Wikipedia [34, 51] in the future. We have collected some GitHub data, and also find the converging W-T patterns there. However, the correlation between W-T patterns and productivity of developers need to be further validated, since currently we haven’t yet collected the data about code length added or deleted in a particular commit. We will show the extended results in our future work.

We acknowledge that, like many other empirical researches, our work is based on a sample of work and talk activities of developers, but not all of them. We just consider the commits to code or documents as work activities, while in reality developers may have other kinds of work activities which are relatively difficult to be captured from OSS repositories. Besides, there might be other kinds of talk activities too, e.g., the discussion on issue tracking systems, that are not included in this study. In fact, we indeed collect issue tracking data from Jira and Bugzilla [25], and do experiments by including them as talk activities (both opening issue as initializing the discussion and comments). However, we don’t find any result that changes dramatically in this case, indicat-

ing the revealed phenomena are quite robust.

We use only LoC to measure the productivity of a developer, while in fact there are alternative metrics, such as the number of issues fixed [60]. Moreover, there are also metrics about work efficiency, such as the development time of tasks [29], and about the quality of code, such as the number of bugs [3]. We will study the effects of work-talk patterns on these metrics in the future, rather than put them together in this single paper. It’s reasonable to use LoC here, since it has been used extensively to measure the productivity of developers [39].

## 7 Conclusions

In this paper, we demonstrate that work-talk patterns of software developers in a number of OSS communities can be effectively studied using sequence analysis methods on sequences arising from simple two-state behavior models. Our methods enabled us to learn about a series of interesting task-oriented community based phenomena: that developers in a community present similar W-T patterns, and this clustering of W-T patterns is enhanced with time, reflecting different work cultures in

these communities, with emphasis on different proportions of continuous work to continuous talk activities; that social and technical interactions may play a role in synchronizing W-T patterns, since developers with stronger social or technical links in a community have more similar W-T patterns; and that although successful task-communities may have relatively different cultures, developers with balanced work-talk patterns seems to play critical roles in sustaining them, and, at least in the ones we studied, each has at least one such developer.

These findings suggest that online individuals may synchronize their behaviors with others to better fit in the task communities and to improve coordinating efficiencies. The methods proposed in this paper can be further expanded and applied to analyze the switching pattern of more varied kinds of activities in more diverse online communities.

## 8 Acknowledgments

The authors would like to thank all the members in our research group in the Department of Computer Science, University of California in Davis, for the valuable discussion about the ideas and technical details presented in this paper. All authors gratefully acknowledge support from the Air Force Office of Scientific Research, award FA955-11-1-0246. QX acknowledges support from the National Natural Science Foundation of China (Grant Nos. 61004097, 61273212) and the China Scholarship Council (CSC).

## References

- [1] A. Abbott. Sequence analysis: new methods for old ideas. *Annual review of sociology*, 21(1):93–113, 1995.
- [2] A. Abbott and A. Hrycak. Measuring resemblance in sequence data: An optimal matching analysis of musicians’ careers. *American journal of sociology*, 96(1), 1990.
- [3] M. Aberdour. Achieving quality in open-source software. *IEEE Software*, 24(1):58–64, 2007.
- [4] P. D. Adamczyk and B. P. Bailey. If not now, when?: the effects of interruption at different moments within task execution. In *Proceedings of the 2004 SIGCHI Conference on Human Factors in Computing Systems*, pages 271–278. ACM, 2004.
- [5] S. Aisenbrey and A. E. Fasang. New life for old ideas: The” second wave” of sequence analysis bringing the” course” back into the life course. *Sociological Methods & Research*, 38(3):420–462, 2010.
- [6] J. H. Astrachan. Family firm and community culture. *Family Business Review*, 1(2):165–189, 1988.
- [7] K. A. Basoglu, M. A. Fuller, and J. T. Sweeney. Investigating the effects of computer mediated interruptions: An analysis of task characteristics and interruption frequency on financial performance. *International Journal of Accounting Information Systems*, 10(4):177–189, 2009.
- [8] C. Bird, A. Gourley, P. Devanbu, M. Gertz, and A. Swaminathan. Mining email social networks. In *Proceedings of the 2006 International Workshop on Mining Software Repositories*, pages 137–143. ACM, 2006.
- [9] C. Bird, A. Gourley, P. Devanbu, A. Swaminathan, and G. Hsu. Open borders? immigration in open source projects. In *Proceedings of the 4th International Working Conference on Mining Software Repositories*, pages 6–6. IEEE, 2007.
- [10] A. V. Bodapati. Recommendation systems with purchase data. *Journal of Marketing Research*, 45(1):77–93, 2008.
- [11] S. P. Borgatti, A. Mehra, D. J. Brass, and G. Labianca. Network analysis in the social sciences. *science*, 323(5916):892–895, 2009.
- [12] C. D. Brummitt, R. M. DSouza, and E. Leicht. Suppressing cascades of load in interdependent networks. *Proceedings of the National Academy of Sciences*, 109(12):E680–E689, 2012.
- [13] C. Brzinsky-Fay, U. Kohler, and M. Luniak. Sequence analysis with stata. *Stata Journal*, 6(4):435, 2006.
- [14] S. V. Buldyrev, R. Parshani, G. Paul, H. E. Stanley, and S. Havlin. Catastrophic cascade of failures in interdependent networks. *Nature*, 464(7291):1025–1028, 2010.
- [15] D. Centola. The spread of behavior in an online social network experiment. *science*, 329(5996):1194–1197, 2010.
- [16] A. Y. Chen, R. B. Sawyers, and P. F. Williams. Reinforcing ethical decision making through corporate culture. *Journal of Business Ethics*, 16(8):855–865, 1997.
- [17] J. P. Cothrel. Measuring the success of an online community. *Strategy & Leadership*, 28(2):17–21, 2000.

- [18] D. R. Cox et al. Regression models and life tables. *JR stat soc B*, 34(2):187–220, 1972.
- [19] L. Dabbish, C. Stuart, J. Tsay, and J. Herbsleb. Social coding in GitHub: transparency and collaboration in an open software repository. In *Proceedings of the ACM 2012 Conference on Computer Supported Cooperative Work*, pages 1277–1286. ACM, 2012.
- [20] B. J. Dempsey, D. Weiss, P. Jones, and J. Greenberg. Who is an open source software developer? *Communications of the ACM*, 45(2):67–72, 2002.
- [21] S. R. Eddy. Hidden markov models. *Current opinion in structural biology*, 6(3):361–365, 1996.
- [22] R. C. Elandt-Johnson and N. L. Johnson. *Survival models and data analysis*. Wiley New York, 1980.
- [23] A. Etzion. *The Active Society: A Theory of Societal and Political Processing*. Free Press, 1968.
- [24] J. H. Fowler and N. A. Christakis. Cooperative behavior cascades in human social networks. *Proceedings of the National Academy of Sciences*, 107(12):5334–5338, 2010.
- [25] M. Gharehyazie, D. Posnett, and V. Filkov. Social activities rival patch submission for prediction of developer initiation in oss projects. In *Proceedings of the 29th International Conference on Software Maintenance*, pages 340–349. IEEE, 2013.
- [26] R. Guimerà, B. Uzzi, J. Spiro, and L. A. N. Amaral. Team assembly mechanisms determine collaboration network structure and team performance. *Science*, 308(5722):697–702, 2005.
- [27] C. Gutwin, R. Penner, and K. Schneider. Group awareness in distributed software development. In *Proceedings of the 2004 ACM Conference on Computer Supported Cooperative Work*, pages 72–81. ACM, 2004.
- [28] J. D. Herbsleb and R. E. Grinter. Architectures, coordination, and distance: Conway’s law and beyond. *IEEE software*, 16(5):63–70, 1999.
- [29] J. D. Herbsleb and J. A. Roberts. Collaboration in software engineering projects: A theory of coordination. In *Proceedings of the 2006 International Conference on Information Systems*, page 38, 2006.
- [30] R. F. Hurley. Group culture and its effect on innovative productivity. *Journal of Engineering and Technology Management*, 12(1):57–75, 1995.
- [31] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: a review. *ACM computing surveys (CSUR)*, 31(3):264–323, 1999.
- [32] N. Kashtan and U. Alon. Spontaneous evolution of modularity and network motifs. *Proceedings of the National Academy of Sciences*, 102(39):13773–13778, 2005.
- [33] R. Kitamura, C. Chen, R. M. Pendyala, and R. Narayanan. Micro-simulation of daily activity-travel patterns for travel demand forecasting. *Transportation*, 27(1):25–51, 2000.
- [34] A. Kittur, B. Suh, B. A. Pendleton, and E. H. Chi. He says, she says: conflict and coordination in wikipedia. In *Proceedings of the 2007 SIGCHI conference on Human factors in computing systems*, pages 453–462. ACM, 2007.
- [35] B. Koerin. Violent crime: Prediction and control. *Crime & Delinquency*, 24(1):49–58, 1978.
- [36] T. D. LaToza, G. Venolia, and R. DeLine. Maintaining mental models: a study of developer work habits. In *Proceedings of the 28th International Conference on Software Engineering*, pages 492–501. ACM, 2006.
- [37] H. Lütkepohl. *New introduction to multiple time series analysis*. Springer, 2007.
- [38] R. S. Mano and G. S. Mesch. E-mail characteristics, work performance and distress. *Computers in Human Behavior*, 26(1):61–69, 2010.
- [39] A. Mockus, R. T. Fielding, and J. D. Herbsleb. Two case studies of open source software development: Apache and mozilla. *ACM Transactions on Software Engineering and Methodology*, 11(3):309–346, 2002.
- [40] M. E. Newman, S. Forrest, and J. Balthrop. Email networks and the spread of computer viruses. *Physical Review E*, 66(3):035101, 2002.
- [41] J.-P. Onnela, J. Saramäki, J. Hyvönen, G. Szabó, D. Lazer, K. Kaski, J. Kertész, and A.-L. Barabási. Structure and tie strengths in mobile communication networks. *Proceedings of the National Academy of Sciences*, 104(18):7332–7336, 2007.
- [42] D. S. Pattison, C. A. Bird, and P. T. Devanbu. Talk and work: a preliminary report. In *Proceedings of the 5th International Working Conference on Mining Software Repositories*, pages 113–116. ACM, 2008.
- [43] J. Preece, B. Nonnecke, and D. Andrews. The top five reasons for lurking: improving community experiences for everyone. *Computers in human behavior*, 20(2):201–223, 2004.

- [44] C. M. Ridings and D. Gefen. Virtual community attraction: Why people hang out online. *Journal of Computer-Mediated Communication*, 10(1):00–00, 2004.
- [45] M. S. Rosenberg. *Sequence alignment: methods, models, concepts, and strategies*. Univ of California Press, 2009.
- [46] A. Sarma, L. Maccherone, P. Wagstrom, and J. Herbsleb. Tesseract: Interactive visual exploration of socio-technical relationships in software development. In *Proceedings of the 31st International Conference on Software Engineering*, pages 23–33. IEEE, 2009.
- [47] K. Sharma. *Bioinformatics: Sequence Alignment and Markov Models*. McGraw-Hill Professional, 2008.
- [48] R. D. C. Team et al. R: A language and environment for statistical computing. *R foundation for Statistical Computing*, 2005.
- [49] R. van Solingen, E. Berghout, and F. van Latum. Interrupts: just a minute never is. *IEEE software*, 15(5):97–103, 1998.
- [50] B. Vasilescu, V. Filkov, and A. Serebrenik. Stackoverflow and github: associations between software development and crowdsourced knowledge. In *Proceedings of the 2013 International Conference on Social Computing*, pages 188–195. IEEE, 2013.
- [51] F. B. Viegas, M. Wattenberg, J. Kriss, and F. Van Ham. Talk before you type: Coordination in wikipedia. In *Proceedings of the 40th Annual Hawaii International Conference on System Sciences*, pages 78–78. IEEE, 2007.
- [52] G. Von Krogh, S. Spaeth, and K. R. Lakhani. Community, joining, and specialization in open source software innovation: a case study. *Research Policy*, 32(7):1217–1241, 2003.
- [53] J. D. Watson and F. H. Crick. The structure of dna. In *Cold Spring Harbor Symposia on Quantitative Biology*, volume 18, pages 123–131. Cold Spring Harbor Laboratory Press, 1953.
- [54] S. Weber. *The success of open source*, volume 368. Cambridge Univ Press, 2004.
- [55] S. Wuchty, B. F. Jones, and B. Uzzi. The increasing dominance of teams in production of knowledge. *Science*, 316(5827):1036–1039, 2007.
- [56] Q. Xuan, F. Du, L. Yu, and G. Chen. Reaction-diffusion processes and metapopulation models on duplex networks. *Physical Review E*, 87(3):032809, 2013.
- [57] Q. Xuan and V. Filkov. Building it together: Synchronous development in OSS. In *Proceedings of the 2014 International Conference of Software Engineering*. ACM, 2014.
- [58] Q. Xuan, M. Gharehyazie, P. T. Devanbu, and V. Filkov. Measuring the effect of social communications on individual working rhythms: A case study of open source software. In *Proceedings of the 2012 International Conference on Social Informatics*, pages 78–85. IEEE, 2012.
- [59] D. Zelikovich. The negative effect of e-mails at work. *Review of International Comparative Management*, 12(3):575–585, 2011.
- [60] M. Zhou and A. Mockus. What make long term contributors: Willingness and opportunity in oss community. In *Proceedings of the 2012 International Conference on Software Engineering*, pages 518–528. IEEE, 2012.

## 9 Appendix

### 9.1 Hidden Markov Model

An HMM is a simple modeling mechanism to explain transitions among several different states. We use an HMM with two states, “work” and “talk”, and transitions between them corresponding to either continuing to perform the same activity, W followed by a W or T followed by a T, or switching activities, W followed by a T, and vice versa. The HMM diagram is shown in Figure 2. If we denote by  $P_W(k)$  and  $P_T(k)$  the probabilities that work, resp. talk, happen at time step  $k$ , then for the next time point we have

$$P_W(k+1) = \alpha P_W(k) + (1-\beta)P_T(k), \quad (10)$$

$$P_T(k+1) = (1-\alpha)P_W(k) + \beta P_T(k). \quad (11)$$

where  $\alpha$  and  $\beta$  are the transition probabilities. We note here that while  $\alpha$  and  $\beta$  could evolve with time, they don’t change much between successive activities, therefore we can consider them as constants in the sequences with certain lengths.

Equations (10) and (11) can be approximated for continuous time,  $\tau$ , and then transformed to the following more compact matrix form:

$$\dot{P}(\tau) = \begin{bmatrix} \alpha - 1 & 1 - \beta \\ 1 - \alpha & \beta - 1 \end{bmatrix} P(\tau), \quad (12)$$

with  $P(\tau) = [P_W(\tau), P_T(\tau)]^T$ . By solving equation (12), we have

$$P(\tau) = C_1 \begin{bmatrix} 1 - \beta \\ 1 - \alpha \end{bmatrix} + C_2 \begin{bmatrix} 1 \\ -1 \end{bmatrix} e^{(\alpha+\beta-2)\tau}. \quad (13)$$



The fractions of work and talk activities,  $P_W$  and  $P_T$ , in a sequence with length  $L$  can be estimated by

$$\begin{bmatrix} P_W \\ P_T \end{bmatrix} = \frac{1}{L} \int_0^L P(\tau). \quad (14)$$

By substituting equation (13) into (14), we have

$$\begin{bmatrix} P_W \\ P_T \end{bmatrix} = \frac{C_1}{(2 - \alpha - \beta)L} \begin{bmatrix} 1 \\ -1 \end{bmatrix} \left[ 1 - e^{(\alpha + \beta - 2)L} \right] + C_2 \begin{bmatrix} 1 - \beta \\ 1 - \alpha \end{bmatrix}. \quad (15)$$

In the right side of equation (15), the first term is negligible when the sequence is long enough, considering  $\alpha + \beta < 2$ . Since it is always satisfied  $P_W + P_T = 1$ , we have

$$P_W = \frac{1 - \beta}{2 - \alpha - \beta}, \quad P_T = \frac{1 - \alpha}{2 - \alpha - \beta}, \quad (16)$$

which are fully determined by the two parameters in the model. Then, the probabilities for the four different two-patterns in the sequence, in terms of  $\alpha$  and  $\beta$ , are given by:

$$P_{WW} = \alpha P_W = \frac{\alpha(1 - \beta)}{2 - \alpha - \beta}, \quad (17)$$

$$P_{WT} = (1 - \alpha) P_W = \frac{(1 - \alpha)(1 - \beta)}{2 - \alpha - \beta}, \quad (18)$$

$$P_{TW} = (1 - \beta) P_T = \frac{(1 - \alpha)(1 - \beta)}{2 - \alpha - \beta}, \quad (19)$$

$$P_{TT} = \beta P_T = \frac{(1 - \alpha)\beta}{2 - \alpha - \beta}, \quad (20)$$

Intuitively, larger  $\alpha$  and  $\beta$  means higher proportions of WW and TT patterns, respectively, in the sequence. Furthermore, the probabilities for longer patterns can be calculated similarly, once the model parameters  $\alpha$  and  $\beta$  are estimated from

equations (17) to (20). It is important to note that we always have  $\alpha + \beta = 1$  in the randomized W-T sequences generated by the null model. In this case,  $\alpha$  and  $\beta$  are equal to the fractions of work and talk activities, respectively.

## 9.2 Hazard Model

Survival analysis enables modeling of outcomes in the presence of censored data. In our case the censoring is due to the uncertainty that long time periods without activities may or may not indicate a developer has left the community. Generally, survival analysis involves calculating the Hazard rate, defined as the limit of the number of events per  $\delta t$  time divided by the number at risk, as  $\delta t \rightarrow 0$ . Here, suppose a developer does not leave the community until time  $\Gamma$ , the Hazard rate is given by

$$h(t) = \lim_{\delta t \rightarrow 0} \frac{P(t \leq \Gamma < t + \delta t | t \leq \Gamma)}{\delta t}. \quad (21)$$

Our primary interest is the survival function defined as  $S(t) = P(t < \Gamma)$ , which can be calculated from equation (21) by

$$S(t) = e^{-\int_0^t h(\tau) d\tau}. \quad (22)$$

Suppose there are several factors, denoted by  $x_i, i = 1, 2, \dots, \gamma$ , that can influence the survival time, then we adopt the Cox model [18] to define the Hazard rate  $h(t)$  by

$$h(t) = h_0(t) e^{\sum_{i=1}^{\gamma} b_i x_i}, \quad (23)$$

with  $h_0(t)$  describing how the hazard changes over time at baseline levels of covariates. Here we focus on the hazard ratio  $h(t)/h_0(t)$  to see whether increasing some covariates will significantly increase or decrease the survival time, e.g.,  $b_i > 0$  means that the individuals of larger  $x_i$  will have statistically shorter survival times.